SELF-DESTRUCTING DOCUMENT AND E-MAIL MESSAGING SYSTEM

Patent number:	WO9858321
Publication date:	1998-12-23
Inventor:	KAPPEL CARY S (US); BAKER STUART D (US); RIES WILLIAM (US); SHERMAN GREG M (US); UDELL HOWARD R (US)
Applicant:	KAPPEL CARY S (US); BAKER STUART D (US); RIES WILLIAM (US); SHERMAN GREG M (US); UDELL HOWARD R (US); PURDUE PHARMA LP (US)
Classification:	
- international:	G06F15/00; G06F17/30
- european:	G06F17/60A2
Application number:	WO1998US12557 19980616
Priority number(s):	US19970049853P 19970617

Also published as:

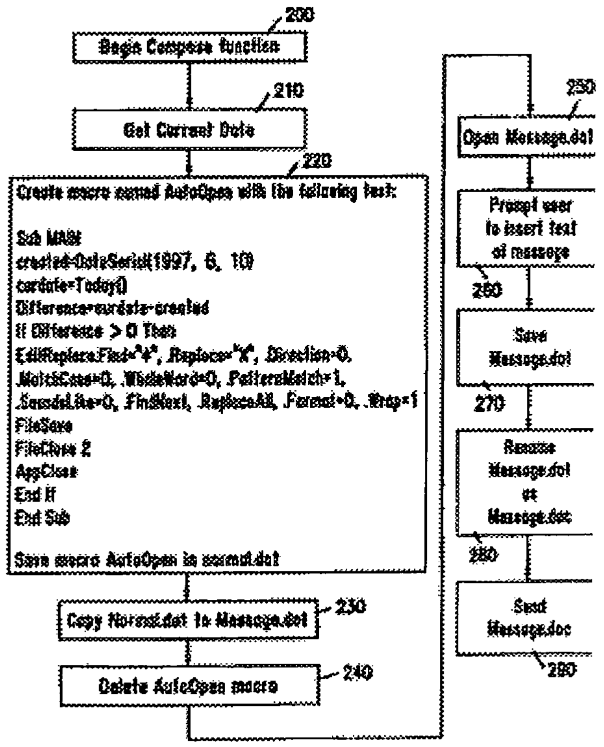
CA2362716 (A
BR9806000 (A)
AU725944 (B2)
AT4261U (U2)

Cited documents:

US5623600
US5781901
US5805702
XP002914521

Abstract of **WO9858321**

A self-destructing document or e-mail messaging system is provided that automatically destroys documents or e-mail messages at a predetermined time by attaching a macro or virus to the document or e-mail message. A macro is created (220) and is attached (230) to file such as an e-mail message (280) or document (270) when it is created. The macro contains a portion of executable code or an executable program which instructs the computer to overwrite and/or delete the file to which the virus is attached at a desired time.



```
Create macro named AutoOpen with the following text:

Sub MAIN
created=DateSerial(1997, 6, 10)
curdate=Today()
Difference=curdate-created
If Difference > 0 Then
EditReplace.Find="*", .Replace="X", .Direction=0,
.MatchCase=0, .WholeWord=0, .PatternMatch=1,
.SoundsLike=0, .FindNext, .ReplaceAll, .Format=0, .Wrap=1
FileSave
FileClose 2
AppClose
End If
End Sub

Save macro AutoOpen in normal.dot
```

host program. In contrast "prepending viruses" attach themselves to the beginning of the host program. Other types of viruses are located in interior portions of the host program. Another classes of viruses are known as "macro" viruses. These viruses are macros embedded in text documents which can be configured to execute whenever the document is opened, created, or saved. Typically, the term Trojan horse is used to refer to a virus which remains with its host file or program, and does not travel to other files or programs.

In accordance with the first embodiment of the present invention, an executable module in the form of a Trojan horse is attached to a file (such as an e-mail message or document) when it is created. The executable module contains a portion of executable code or an executable program which instructs the computer to overwrite and/or delete the file to which the executable module is attached at a desired time. In this manner, the executable module sets a limit on the lifetime of the document or e-mail message.

Since the executable module is attached to the file, it will travel with the file even when the file is copied, forwarded, or saved to disks or tape drives.

In accordance with a further aspect of the first embodiment of the invention, the executable module executes each time that a file to which it is attached is opened. The executable module determines whether a condition precedent to file deletion has been met. If the condition precedent is met, the executable module instructs the computer to overwrite the file with null data, and then to save, close or delete the file.

In this manner, the present invention provides a number of advantages over prior art systems. For example, since the executable module is attached to the file and is executed whenever the file is opened, the system according to the present invention can enforce a document retention policy even if the file has been stored on external media such as floppy disks, or on a non-network computer. This is because the executable module travels with the file, and is executed whenever, and wherever, the file is opened.

In accordance with a second embodiment of the present invention, a self destructing email messaging system is provided which automatically attaches an executable module to each e-mail message or e-mail message attachment which is created. The executable module contains a portion of executable code or an executable program which instructs the computer to overwrite and/or delete the message (and/or message attachment) to which the executable module is attached at a desired time. The executable module travels with the message as the message is transmitted to its addressee because the executable module is attached to the message. Moreover, the executable module remains attached to the message even when the message is copied, forwarded to another addressee, or saved to disks or other external media.

In accordance with a third embodiment of the present invention, a document security system includes a virtual container into which one or more digital objects are "placed."
In this regard, the term "digital object" is used broadly, and includes documents (such as spreadsheets, charts/graphs, word processor documents, ASCII text files, images, and other data files), programs, and anything else which can be stored on a computer.

The document security system includes a container creator and a container opener. The container creator and the container opener are implemented in one or more software programs which are executed on a computer. The container opener allows a user to set a limit on the valid lifetime of the digital objects in the container. These "lifetime controls" may include, for example, an expiration date, an expiration date and time, a fixed number of times in which the document can be opened, or other limits. When the container is opened by the container opener, the container opener checks the lifetime controls. If they are valid, the digital objects contained within the container are accessible for display or execution by the user. If, however, the lifetime controls are invalid, the container opener will immediately destroy the digital object.

An advantage of the document security system including the container creator and the container opener is that, in certain environments, it provides a more robust enforcement of lifetime controls than a self-destructing document containing an executable module.

This is because the ability of current word processing and spreadsheet programs to execute an executable module varies widely from product to product. Therefore, for example, a self-destructing document which includes an executable module which is executable by MicrosoftWord will not self destruct if opened, for example, as a simple ASCII file which is not configured to recognize and execute the executable module. In contrast, with the document security system, the documents are secured within a container which is opened by the container opener, and the container opener, when used, can enforce the lifetime controls against any document in the container, regardless of its type.

In accordance with a further aspect of the document security system in accordance with the third embodiment of the present invention, each digital object within the container may have independent lifetime controls. In accordance with this feature, when a single object within the container expires, it can be destroyed, while the other objects remain intact.

In accordance with still another embodiment of the document security system, the digital object is a self-destructing document created in accordance with the first or second embodiments of the present invention.

In accordance with a still further aspect of this embodiment, the container, its contents, and its lifetime controls can be secured against a user who wishes to subvert the security system. This security is effected through the use of encryption technology.

Specifically, the container creator is configured to encrypt the objects within the container, and the container opener is configured to decrypt the objects. In this manner, if a user opens one of the objects without utilizing the container opener, the object will be unreadable.

In accordance with a further aspect of the first and second embodiments, the selfdestructing document or e-mail message is encrypted, either by the executable module or by another utility, and the executable module is configured to decrypt the document or e-mail message only if the lifetime of the document or e-mail message has not expired.

In accordance with another embodiment of the present invention, a self-destructing document is created by embedding a plurality of executable modules into a document or e-mail message, wherein each module is executable by a different word processing or e-mail system. For example, a document could include a first module which is executable by a first system and a second module which is executable by a second system. The document itself could be a document native to either system. In accordance with this embodiment, the lifetime controls for the document will be enforced regardless of whether it is opened in the first system or the second system.

In accordance with another embodiment of the invention, an Internet commerce system is provided which employs the virtual containers. In accordance with this embodiment, a party who wishes to sell an electronically transmittable product over the Internet places the product into a virtual container using a container creator utility which encrypts the product and sets lifetime controls for the product. A potential buyer of the product who wishes to sample the product prior to purchasing it, obtains a copy of the container along with an container opener utility from the seller. The container opener utility allows the potential buyer to view and use the product only until a preselected expiration date. The container opener does not, in this embodiment, allow the user to remove the product from the virtual container. In accordance with a preferred aspect of this embodiment, the product is deleted by the container opener if the container is opened after the expiration date. In accordance with yet another aspect of this embodiment, if the seller receives payment for the product prior to the expiration date, the seller will transmit a unique key to the buyer. The container opener could be configured to release the product from the container upon receiving the key.]

Brief Description of the Drawings

Figure 1 shows an illustrative prior art environment in which the present invention can be implemented.

Figures 2(a) and 2(b) show e-mail messages in accordance with the present invention which include pre-pending and appending viruses, respectively.

Figure 3 is a flow chart of an illustrative method for creating a self-destructing e-mail message in accordance with an illustrative embodiment of the present invention.

Figure 4 is a flow chart of a macro virus which is embedded in the self-destructing email message of Figure 3.

Figures 5(a) through 5(c) illustrate a graphical user interface for a self-destructing document in accordance with an embodiment of the present invention which is implemented for a MicrosoftExcel document.

Figures 6(a) through 6(c) illustrate a graphical user interface for a self-destructing document in accordance with an embodiment of the present invention which is implemented for a MicrosoftWordTM document.

Figures 7(a) through 7(e) illustrate a graphical user interface for a self-destructing email message in accordance with an embodiment of the present invention which is implemented in Microsoft OutlookTM.

ofoverwriteldelete conditions. This could be implemented, for example, through a tools menu.

An exemplary method for implementing a self-destructing electronic document or providing a self-destructing electronic messaging system will now be described. While the illustrative embodiment described herein is illustrated for Microsoft WordTM,
Microsoft ExcelTM, and MicrosoftOutlookTM environments, one of ordinary skill in the art will appreciate that the present invention may be implemented in a variety of environments and in a variety of ways.

Figures 3 and 4 illustrate the implementation of a self-destructing electronic messaging system for MicrosoftWord 6.0 documents. Referring to Figure 3, at step 200, a user initiates the creation of a document or message on, for example, office computer 10.1 of Figure 1. At step 220, the system creates an "AutoOpen" macro and saves the macro in the "Normal.dot" file. In accordance with the MicrosoftWord architecture, a macro created with the title AutoOpen will execute each time a Word document is opened. The instructions referenced at step 220, which are written in the WordBasicTM programming language, are saved as an AutoOpen macro in the normal.dot template file. Normal.dot is a template which is designated by theWord program as a storage file for global macros. At step 230, the normal.dot file, which includes the AutoOpen macro, is copied to a file named "message.dot." Then, at step 240, the AutoOpen macro is deleted from the normal.dot file. At steps 250-260, the message.dot file is opened and the user is prompted to insert the text of the document or message. Then, at steps 270, the message. dot file, which now includes the document or message as well as a copy of the normal.dot file, is saved. The copy of the normal.dot file, in turn, includes the AutoOpen macro. At steps 280-290, the message.dot file is renamed message.doc, and then sent as an electronic mail message or electronic mail message attachment.

[The electronic mail message is sent, for example, over the Internet to Home Computer 40 via LAN server 20 and Internet servers 30. The AutoOpen macro, which is embedded in the message.doc file, will execute when the file message.doc is opened by the addressee of the message. Figure 4 is a flow chart for the AutoOpen macro which is embedded in the message.doc file. At step 310, the DateSerial() function returns a serial value representative of the creation date of the message as the variable "created."
The argument for the function DateSerial is set at the time the AutoOpen macro is created (steps 210 and 220 of Figure 3), and has been arbitrarily indicated as June 10, 1997 in Figures 3 and 4. At step 320, the Today() function returns a serial value representative of the current date as the variable "curdate". The serial values returned by these functions are integers between 1 and 802074, with 1 corresponding toDecember 31, 1899 and 802074 corresponding toDecember 31, 4095. At step 330, the variable "difference" is set as the difference between the "curdate" and "created."
Consequently, the variable "difference" is equal to the number of days which have elapsed between the creation of the message.doc file and the current date in which the document has been opened.

Steps 340 through 360 constitute an "If, Then" Statement which determines whether the time elapsed between the creation date and the current date has exceeded a predetermined threshold. For purposes of illustration, the threshold has been set at 60 days. At step 340, the variable "difference" is compared to the value 60. If "difference" is greater than 60, i.e., if more than 60 days have elapsed since the message.doc file was created, then steps 350, 360, and 370 are executed. At step 350, an EditReplace.Find function is performed. This function will replace each character of the message.doc file with a null character (which has been arbitrarily set as "X".) Then, at step 360, the message.doc file, which has been rewritten as a series of"X" characters equal to the number of text characters in the file, is saved and closed. In this manner, the memory locations from which the message.doc file was retrieved are overwritten with null characters. In contrast, if, at step 340, the variable "Difference" is less than or equal to 60, then execution jumps to step 370 and the addressee of the message is able to freely view and edit the message.

Naturally, the macro shown in Figure 4 can be modified to perform additional functions. For example, the macro could be modified to delete the message.doc file after it is saved by utilizing the FileNameFromWindow$() function to retrieve the location of the message.doc file at the time it is opened, and then utilizing the Kill function to delete the file. In addition, the macro could be modified to allow the user who is opening the message.doc file to defer message deletion.]

Figures 5(a) through 5(c) and Table 1, infra, show a self-destructing document system which is illustrated with regard to MicrosoftExcel documents. Referring to Table 1, an illustrative VisualBasic program for creating a self-destructing Excel document is shown which includes a user interface implemented for a Microsoft WindowsTM environment. The implementation shown utilizes the "auto~open" and "auto~close" routine supported by MicrosoftExcel

```
Range("AI ").Select
Range(Selection, Selection.SpecialCells(xlLastCell)).Select
Selection.Clear
ActiveWorkbook.Save
```

Figures 6a through 6c and Tables 2(a,b) show a self-destructing document system which is illustrated with